International Journal of Approximate Reasoning 50 (2009) 1129–1147

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

journal homepage: www.elsevier.com/locate/ijar

A new probabilistic fuzzy model: Fuzzification–Maximization (FM) approach

Sungjun Hong, Heesung Lee, Euntai Kim *

School of Electrical and Electronic Engineering, Yonsei University, C613, Sinchon-dong, Seodaemun-gu, Seoul 120-749, Republic of Korea

ARTICLE INFO

Article history:
Received 19 July 2008
Received in revised form 7 May 2009
Accepted 8 May 2009
Available online 21 May 2009

Keywords:
Probabilistic fuzzy model
Robust learning
Noise
Coarse tuning
Fine tuning
Fuzzification–Maximization

ABSTRACT

Over the past few decades, fuzzy logic systems have been used for nonlinear modeling and approximation in many fields ranging from engineering to science. In this paper, a new fuzzy model is developed from the probabilistic and statistical point of view. The proposed model decomposes the input–output characteristics into noise-free part and probabilistic noise part and identifies them simultaneously. The noise-free model recovers the nominal input–output characteristics of the target system and the noise model gives approximation to the probabilistic nature of the added noise. To identify the two submodels simultaneously, we propose the Fuzzification–Maximization (FM). Finally, some simulations are conducted and the effectiveness of the proposed method is demonstrated through the comparison with the previous methods.

1. Introduction

Nonlinear modeling of complex irregular systems constitutes the essential part of many control and decision making systems. Over the past few decades the intelligent algorithms are used for the purpose especially in cases where not only the internal parameters but also the structure of the target system is unknown. The fuzzy systems used for the purpose are called as fuzzy models.

The major strength of the fuzzy model over other intelligent models lies in its transparency and interpretability and much research has been conducted in that direction: for example, binary logic optimization method was used to build an interpretable model in [1] and linguistic modeling was employed in [2,3]. Singleton consequents were utilized in [4–6] and the fuzzy system equipped with the learning capability of the neural network called SOFIN was proposed in [7,8]. Some interesting works were reported about the balance or trade-off between the interpretability and accuracy [9–12].

As another research direction in fuzzy model, a tremendous amount of work has been conducted towards only the accuracy while sacrificing the transparency and interpretability. The most representative work in this direction is TSK fuzzy model [13–15] and its variants [16–23].

Recently, some interesting works were reported about the robust or probabilistic fuzzy models. The works are different from [1–23] in that the sample data points are assumed to be contaminated with noise and a fuzzy model is built in the presence of the measurement noise. For example, the outliers were removed based on confidence intervals of the residual in [24] and by the robust fuzzy regression agglomeration (RFRA) and the robust learning theory in [25,26]. The relevance vector

* Corresponding author.
E-mail address: etkim@yonsei.ac.kr (E. Kim).

0888-613X/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.ijar.2009.05.004

machine (RVM) was employed to build a robust fuzzy model in [27]. The distribution of the samples was considered and a model was built from the probabilistic and statistical point of view in [28,29]. They were also linguistically interpretable.

In this paper, a new fuzzy model is developed from the probabilistic and statistical point of view. The new model decomposes the input–output characteristics into noise-free part and probabilistic noise part and identifies them simultaneously. The noise-free model recovers the nominal input–output characteristics of the target system and the noise model gives approximation to the probabilistic nature of the added noise. To identify the two submodels simultaneously, we propose the Fuzzification–Maximization (FM) which is inspired by the well known Expectation–Maximization (EM) method [30,31]. The proposed probabilistic model has three features:

(1) The proposed model recovers not only the noise-free input–output relation of the unknown system but also captures the probabilistic characteristics of the added noise. *That is, the probabilistic nature of the noise is also approximated by a fuzzy model.*
(2) The model parameters are identified not by minimizing the modeling error but by maximizing the likelihood from probabilistic point of view. Thus, the resulting fuzzy model becomes automatically robust against outliers.
(3) In many applications in control and signal processing such as optimal filtering (Kalman filtering or particle filtering [30]), we need not only the noise-free model but also the model for noise.

The remainder of this paper is organized as follows: In Section 2, the previous deterministic fuzzy modeling methods are reviewed briefly and a new probabilistic fuzzy model is proposed. In Section 3, the proposed probabilistic fuzzy model is identified through coarse tuning and fine tuning. For coarse tuning, the FM is proposed. For fine tuning, the overall likelihood is maximized by the gradient ascent. In Section 4, some experiments are performed to show the validity of the proposed fuzzy model. Some conclusions are drawn in Section 5.

2. Preliminaries fundamentals and problem formulation

Assume that we are given a set of N sample data $S = \{\mathbf{x}_i, y_i\}_{i=1}^N$ and we try to build a fuzzy model such that the data set S is represented as well as possible by the fuzzy model.

2.1. Deterministic TSK fuzzy model

2.1.1. Model

The fuzzy model suggested by Takagi et al. can represent or model a general class of nonlinear systems [13–15]. It is based on fuzzy partition of input space and it can be viewed as the expansion of piecewise linear partition as in (1):

$$R^r: \text{If } x_1 \text{ is } A_1^r, x_2 \text{ is } A_2^r, \ldots, x_d \text{ is } A_d^r, \quad \text{then } y = a_0^r + a_1^r x_1 + \cdots + a_d^r x_d = \mathbf{x}^T \mathbf{a}^r \tag{1}$$

$$y_{fuz}(\mathbf{x}) = \frac{\sum_{r=1}^c \varpi^r y^r(\mathbf{x})}{\sum_{r=1}^c \varpi^r} = \sum_{r=1}^c h^r y^r(\mathbf{x}),$$

where $\mathbf{a}^r = (\,a_0^r \quad a_1^r \quad \cdots \quad a_d^r\,)^T \in \Re^{d+1}$, $\mathbf{x} = (\,1 \quad x_1 \quad x_2 \quad \cdots \quad x_d\,)^T \in \Re^{d+1}$, $y^r(\mathbf{x}) = a_0^r + a_1^r x_1 + \cdots + a_d^r x_d = \mathbf{x}^T \mathbf{a}^r$, $\varpi^r = t_{k=1}^d \times \{A_k^r(x_i)\}$ and $h^r = \frac{\varpi^r}{\sum_{r=1}^c \varpi^r}$. R^r ($r = 1,2,\ldots,c$) denotes the rth fuzzy rule of the fuzzy model and x_j is the jth input variable. $t(\cdot)$ denotes a t-norm for fuzzy implication. $y^r(\mathbf{x})$ is the output of the rth fuzzy rule for an input $\mathbf{x}$. $A_1^r, A_2^r, \ldots, A_d^r$ are the premise membership functions representing fuzzy subspaces in which the implication R^r can be applied for reasoning. The membership functions considered in this paper are assumed to be bell-typed. The physical meaning of the model is that when the input variable $\mathbf{x}$ is constrained to a fuzzy range R^r characterized by the premise parts, the output is a hyperplane-shaped linear function of the input variables. Therefore, the TSK fuzzy system can be viewed as fuzzy blending of the linear system models.

2.1.2. Identification

The original identification algorithm of the fuzzy model reported in [13–15] was so complicated and difficult to implement that it was not widely used. To overcome the problem, the two-phase methods based on clustering were reported [16,17]. The methods usually consist of two-phases: Coarse tuning phase and fine tuning phase as in Fig. 1.

In the coarse tuning phase, a rough partition of the input–output space is made and hyperplane-shaped unsupervised clusters are built. Each cluster plays the role of a fuzzy rule in the TSK fuzzy model. Fig. 2 shows the basic concept of the hyperplane-shaped clustering. In the fine tuning phase, the parameters obtained in the coarse tuning are used as initial parameters and the parameters are finely adjusted such that the difference between the actual output y_i and the fuzzy model output $y_{fuz}(\mathbf{x}_i)$ is minimized. More specifically, the modeling error e is defined as

$$e \triangleq y - y_{fuz}(\mathbf{x}) = y - \frac{\sum_{r=1}^c \varpi^r y^r(\mathbf{x})}{\sum_{r=1}^c \varpi^r} \tag{2}$$

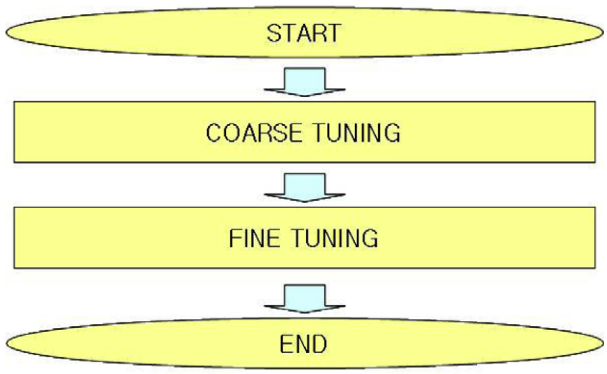


Fig. 1. Two-phase fuzzy modeling.

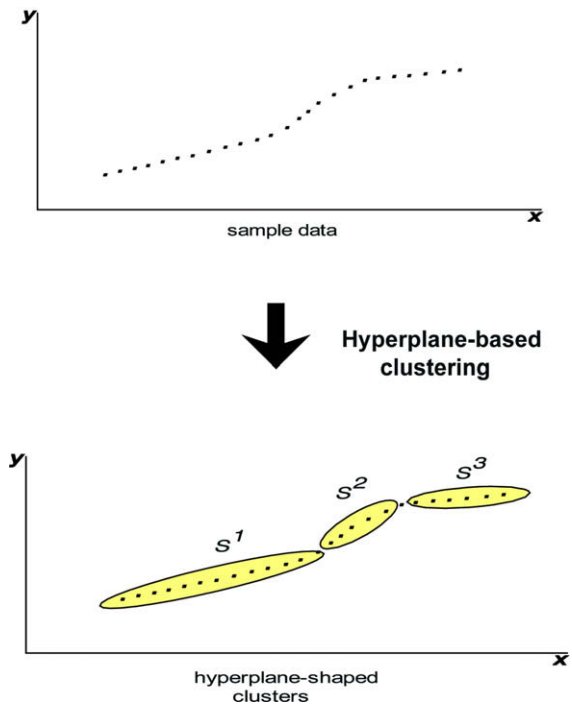


Fig. 2. Hyperplane-shaped clustering in the fuzzy modeling.

and the fuzzy model parameter $\boldsymbol{\Theta}$ is adjusted in the direction of the negative gradient of the squared modeling error by

$$\Delta\boldsymbol{\Theta} = -\gamma \frac{\partial}{\partial \boldsymbol{\Theta}}\left(\frac{e^2}{2}\right) = -\gamma e \frac{\partial e}{\partial \boldsymbol{\Theta}}. \tag{3}$$

2.2. Probabilistic fuzzy model: problem formulation

In this subsection, the probabilistic fuzzy model is proposed and it is formulated as a single Gaussian distribution with an input-dependent variance. The probabilistic model (1) is the extension of the deterministic model and is of the following form:

$$R^r : \text{If } x_1 \text{ is } A_1^r, x_2 \text{ is } A_2^r, \ldots, x_d \text{ is } A_d^r, \quad \text{then } y \to N(y|y^r(\mathbf{x}), (\sigma^r)^2) \tag{4}$$

where $y^r(\mathbf{x}) = a_0^r + a_1^r x_1 + \cdots + a_d^r x_d = \mathbf{x}^T \mathbf{a}^r$ and $(\sigma^r)^2$ is the variance associated with R^r. The basic idea of the probabilistic fuzzy model is that not only the nominal input–output characteristics but also the noise depends on input space and, thus, the noise also should be modeled by the fuzzy model. The fuzzy model (4) can be rewritten as

$$R^r: \text{If } x_1 \text{ is } A_1^r, x_2 \text{ is } A_2^r, \ldots, x_d \text{ is } A_d^r, \quad \text{then } y = a_0^r + a_1^r x_1 + \cdots + a_d^r x_d + n^r = y^r + n^r \tag{5}$$

where $n^r \sim N\left(\cdot|0,(\sigma^r)^2\right)$, that is, $p(n^r) = \frac{1}{\sqrt{2\pi(\sigma^r)^2}} \exp\left\{-\frac{1}{2}\left(\frac{n^r}{\sigma^r}\right)^2\right\}$. The output of the fuzzy model is represented by

$$y_{fuz}(\mathbf{x}) = \frac{\sum_{r=1}^c \varpi^r y^r(\mathbf{x})}{\sum_{r=1}^c \varpi^r} + \frac{\sum_{r=1}^c \varpi^r n^r(\mathbf{x})}{\sum_{r=1}^c \varpi^r} = \sum_{r=1}^c h^r y^r(\mathbf{x}) + \sum_{r=1}^c h^r n^r(\mathbf{x}) \tag{6}$$

where $\varpi^r = t_{k=1}^d \{A_k^r(x_i)\}$, $h^r = \frac{\varpi^r}{\sum_{r=1}^c \varpi^r}$, and $\sum_{r=1}^c h^r = 1$. It is assumed that n^r's are independent of each other. Then the output of the fuzzy model is represented by (4) and (6) as a single Gaussian distribution

$$y_{fuz}(\mathbf{x}) \sim N\left(y|y_{fuzmean}(\mathbf{x}),\ \sigma_{fuz}^2(\mathbf{x})\right) \tag{7}$$

where

$$y_{fuzmean}(\mathbf{x}) = \sum_{r=1}^c h^r y^r(\mathbf{x}) \tag{8}$$

$$\sigma_{fuz}^2(\mathbf{x}) = \sum_{r=1}^c (h^r)^2 (\sigma^r)^2 \tag{9}$$

The mean of the probabilistic fuzzy model is only the deterministic fuzzy model (1) and the probabilistic parameter $\sigma_{fuz}^2(\mathbf{x})$ is also a fuzzy model depending on the input variable $\mathbf{x}$ via $h^r(\mathbf{x})$.

3. Identification of probabilistic fuzzy model

3.1. Coarse tuning

In this subsection, we roughly partition the input–output space and build the hyperplane-shaped clusters as in other two-phase fuzzy modeling methods [16,17]. Unlike the previous deterministic models, however, we do not directly apply the clustering to the sample data. Instead, we consider the likelihood of samples since we are dealing with the probabilistic model. More specifically, we first compute the likelihood of samples under the assumption that the data are fuzzily assigned to hyperplane-shaped clusters and then maximize the likelihood. We alternate two steps (fuzzy assignment and maximization) and we call the strategy as Fuzzification–Maximization (FM) method. The FM is motivated by the Expectation–Maximization (EM) [30,31]. Now, let us derive the FM for probabilistic modeling. First, assume that each data point $(\mathbf{x}_i, y_i)$ $i = 1, 2, \ldots, N$ belongs to one of c hyperplane-shaped clusters *exclusively*. If we denote the set of samples which belong to the rth cluster by S^r, then $S = \{\mathbf{x}_i, y_i\}_{i=1}^N$ can be decomposed into c disjoint subsets S^r, that is $S = \cup_{r=1}^c S^r$ and $S^r \cap S^q = \emptyset$ $(r \neq q)$. For subsequent use, we denote the parameters of the fuzzy model by $\boldsymbol{\Theta}$. If we use the Gaussian membership functions

$$A_k^r(x) = \exp\left\{-\left(\frac{x - p_{k1}^r}{p_{k2}^r}\right)^2\right\} \tag{10}$$

for the premise parts, then the parameter $\boldsymbol{\Theta}$ of the fuzzy model with c fuzzy rules is defined as

$$\boldsymbol{\Theta} = \begin{pmatrix} \boldsymbol{\Theta}^1 & \boldsymbol{\Theta}^2 & \cdots & \boldsymbol{\Theta}^c \end{pmatrix}$$
$$\boldsymbol{\Theta}^r = \Bigg(\underbrace{p_{11}^r \quad p_{12}^r \quad p_{21}^r \quad p_{22}^r \quad \cdots \quad p_{d1}^r \quad p_{d2}^r}_{\text{Premise variables}} \underbrace{a_0^r \quad a_1^r \quad \cdots \quad a_d^r}_{\text{Consequent variables } \mathbf{a}^r} \underbrace{\sigma^r}_{\text{probabilistic variable}} \Bigg). \tag{11}$$

If we define $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ and $\mathbf{Y} = \{y_i\}_{i=1}^N$, then the probability of the outputs $\mathbf{Y}$conditioned on the input $\mathbf{X}$ and fuzzy model $\boldsymbol{\Theta}$ is represented as $p(\mathbf{Y}|\mathbf{X},\boldsymbol{\Theta}) = \prod_{i=1}^N p(y_i|\mathbf{x}_i,\boldsymbol{\Theta})$. We can rewrite the likelihood into

$$p(\mathbf{Y}|\mathbf{X},\boldsymbol{\Theta}) = \prod_{i=1}^N p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}) = \left(\prod_{i\in S^1} p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^1)\right)\left(\prod_{i\in S^2} p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^2)\right)\cdots\left(\prod_{i\in S^c} p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^c)\right) = \prod_{r=1}^c \left(\prod_{i\in S^r} p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^r)\right), \tag{12}$$

where $p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^r)$ is the probability that an output y_i is incurred by an input $\mathbf{x}_i$ when the rth fuzzy rule is activated. Here, we use the classical trick of taking logarithm to maximize the likelihood. By taking the logarithm of (12), we obtain

$$\begin{aligned}\log p(\mathbf{Y}|\mathbf{X},\boldsymbol{\Theta}) &= \log\left(\prod_{r=1}^c \left(\prod_{i\in S^r} p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^r)\right)\right) = \sum_{r=1}^c \sum_{i\in S^r} \log\left(p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^r)\right) = \sum_{r=1}^c \sum_{i=1}^N \left\{I(\mathbf{x}_i \in S^r)\log\left(p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^r)\right)\right\} \\ &= \sum_{i=1}^N \sum_{r=1}^c \left\{I(\mathbf{x}_i \in S^r)\log\left(p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^r)\right)\right\}\end{aligned} \tag{13}$$

where $I(\cdot)$ is the indicator function and (13) is the function that we have to maximize. In reality, however, a sample $(\mathbf{x}_i, y_i)$ does not belong to a single cluster exclusively but fuzzily belongs to all clusters to some degrees. We thus fuzzify the likelihood (13) by replacing the indicator function $I(i \in S^r)$ with the membership functions $h^r(\mathbf{x}_i)$ and allowing each sample $(\mathbf{x}_i, y_i)$ to belong to all clusters to some degrees.

3.1.1. Fuzzification step (F-step)

Let us consider fuzzified likelihood $p_F(\mathbf{Y}|\mathbf{X},\boldsymbol{\Theta})$ defined by $\log p_F(\mathbf{Y}|\mathbf{X},\boldsymbol{\Theta}) = \sum_{i=1}^{N}\sum_{r=1}^{c}\{h^r(\mathbf{x}_i)\log(p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^r))\}$. By the fuzzy model (5), we obtain

$$\begin{aligned}\log p_F(\mathbf{Y}|\mathbf{X},\boldsymbol{\Theta}) &= \sum_{i=1}^{N}\sum_{r=1}^{c}\left\{h^r(\mathbf{x}_i)\log\left(N(y_i|y^r(\mathbf{x}_i),(\sigma^r)^2)\right)\right\}\\ &= \sum_{i=1}^{N}\sum_{r=1}^{c}\left\{h^r(\mathbf{x}_i)\log\left(\frac{1}{\sqrt{2\pi(\sigma^r)^2}}\exp\left(-\frac{1}{2}\frac{(y_i-y^r(\mathbf{x}_i))^2}{(\sigma^r)^2}\right)\right)\right\}\end{aligned} \tag{14}$$

In the F-step, we assume that the consequent parameters $\mathbf{a}^r = (\,a_0^r \quad a_1^r \quad \cdots \quad a_d^r\,)$ in (5) are given and then we can compute $p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^r)$. Thus, each data can be fuzzily assigned to all clusters by

$$h^r(\mathbf{x}_i) = \frac{p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^r)}{\sum_{q=1}^{c}\{p(y_i|\mathbf{x}_i,\boldsymbol{\Theta}^q)\}} = \frac{\left(\frac{1}{\sqrt{2\pi(\sigma^r)^2}}\exp\left(-\frac{1}{2}\frac{(y_i-y^r(\mathbf{x}_i))^2}{(\sigma^r)^2}\right)\right)}{\sum_{q=1}^{c}\left(\frac{1}{\sqrt{2\pi(\sigma^q)^2}}\exp\left(-\frac{1}{2}\frac{(y_i-y^q(\mathbf{x}_i))^2}{(\sigma^q)^2}\right)\right)}. \tag{15}$$

3.1.2. Maximization step (M-step)

In the M-step, we assume that the premise membership values $h^r(\mathbf{x}_i)$ are known and we maximize the fuzzified likelihood function

$$\begin{aligned}\log p_F(\mathbf{Y}|\mathbf{X},\boldsymbol{\Theta}) &= \sum_{i=1}^{N}\sum_{r=1}^{c}\left\{h^r(\mathbf{x}_i)\log\left(\frac{1}{\sqrt{2\pi(\sigma^r)^2}}\exp\left(-\frac{1}{2}\frac{(y_i-y^r(\mathbf{x}_i))^2}{(\sigma^r)^2}\right)\right)\right\}\\ &= \sum_{i=1}^{N}\sum_{r=1}^{c}\left[h^r(\mathbf{x}_i)\left\{-\frac{1}{2}\log\left(2\pi(\sigma^r)^2\right)-\frac{1}{2}\frac{(y_i-y^r(\mathbf{x}_i))^2}{(\sigma^r)^2}\right\}\right].\end{aligned} \tag{16}$$

To maximize the fuzzified likelihood function, we differentiate it with respect to the unknown parameters. First, we differentiate the likelihood with respect to σ^r by

$$\begin{aligned}\frac{\partial}{\partial\sigma^r}\log p_F(\mathbf{Y}|\mathbf{X},\boldsymbol{\Theta}) &= \frac{\partial}{\partial\sigma^r}\sum_{i=1}^{N}\sum_{r=1}^{c}\left[h^r(\mathbf{x}_i)\left\{-\frac{1}{2}\log\left(2\pi(\sigma^r)^2\right)-\frac{1}{2}\frac{(y_i-y^r(\mathbf{x}_i))^2}{(\sigma^r)^2}\right\}\right]\\ &= \sum_{i=1}^{N}\frac{\partial}{\partial\sigma^r}\left[h^r(\mathbf{x}_i)\left\{-\frac{1}{2}\log\left(2\pi(\sigma^r)^2\right)-\frac{1}{2}\frac{(y_i-y^r(\mathbf{x}_i))^2}{(\sigma^r)^2}\right\}\right]\\ &= \sum_{i=1}^{N}\left\{-\frac{h^r(\mathbf{x}_i)}{\sigma^r}+\frac{h^r(\mathbf{x}_i)}{(\sigma^r)^3}(y_i-y^r(\mathbf{x}_i))^2\right\} = -\frac{1}{\sigma^r}\sum_{i=1}^{N}h^r(\mathbf{x}_i)+\frac{1}{(\sigma^r)^3}\sum_{i=1}^{N}\left\{h^r(\mathbf{x}_i)(y_i-y^r(\mathbf{x}_i))^2\right\}.\end{aligned} \tag{17}$$

By setting $\frac{\partial}{\partial\sigma^r}\log p_F(\mathbf{Y}|\mathbf{X},\boldsymbol{\Theta}) = 0$, we obtain

$$\sigma^r = \sqrt{\frac{\sum_{i=1}^{N}\left\{h^r(\mathbf{x}_i)(y_i-y^r(\mathbf{x}_i))^2\right\}}{\sum_{i=1}^{N}h^r(\mathbf{x}_i)}}. \tag{18}$$

Next, we differentiate the likelihood with respect to $\mathbf{a}^r$ as in

$$\begin{aligned}\frac{\partial}{\partial\mathbf{a}^r}\log p_F(\mathbf{Y}|\mathbf{X},\boldsymbol{\Theta}) &= \frac{\partial}{\partial\mathbf{a}^r}\sum_{i=1}^{N}\sum_{r=1}^{c}\left[h^r(\mathbf{x}_i)\left\{-\frac{1}{2}\log\left(2\pi(\sigma^r)^2\right)-\frac{1}{2}\frac{(y_i-y^r(\mathbf{x}_i))^2}{(\sigma^r)^2}\right\}\right]\\ &= \frac{\partial}{\partial\mathbf{a}^r}\sum_{i=1}^{N}\sum_{r=1}^{c}\left[h^r(\mathbf{x}_i)\left\{-\frac{1}{2}\log\left(2\pi(\sigma^r)^2\right)-\frac{1}{2}\frac{(y_i-\mathbf{a}^{rT}\mathbf{x}_i)^2}{(\sigma^r)^2}\right\}\right]\\ &= \sum_{i=1}^{N}\left[-\frac{h^r(\mathbf{x}_i)}{2(\sigma^r)^2}\frac{\partial}{\partial\mathbf{a}^r}(y_i-\mathbf{a}^{rT}\mathbf{x}_i)^2\right] = \frac{1}{(\sigma^r)^2}\sum_{i=1}^{N}\left[h^r(\mathbf{x}_i)(y_i-\mathbf{a}^{rT}\mathbf{x}_i)\mathbf{x}_i\right]\\ &= \frac{1}{(\sigma^r)^2}\left\{\sum_{i=1}^{N}(h^r(\mathbf{x}_i)y_i\mathbf{x}_i)-\sum_{i=1}^{N}(h^r(\mathbf{x}_i)(\mathbf{a}^{rT}\mathbf{x}_i)\mathbf{x}_i)\right\}\\ &= \frac{1}{(\sigma^r)^2}\left\{\sum_{i=1}^{N}(h^r(\mathbf{x}_i)y_i\mathbf{x}_i)-\sum_{i=1}^{N}(h^r(\mathbf{x}_i)\mathbf{x}_i\mathbf{x}_i^T)\mathbf{a}^r\right\}.\end{aligned} \tag{19}$$

By setting $\frac{\partial}{\partial \mathbf{a}^r} \log p_F(\mathbf{Y}|\mathbf{X}, \mathbf{\Theta}) = \mathbf{0}$, we obtain

$$\mathbf{a}^r = \left(\sum_{i=1}^{N} (h^r(\mathbf{x}_i)\mathbf{x}_i\mathbf{x}_i^T) \right)^{-1} \left(\sum_{i=1}^{N} (h^r(\mathbf{x}_i) y_i \mathbf{x}_i) \right). \tag{20}$$

It is known that (20) is the weighted least square solution and can be implemented as the WRLS algorithm given in Appendix. By combining the F-step and M-step, we obtain the following FM-PFCRM. (Fuzzification–Maximization – Probabilistic Fuzzy C Regression Model). The FM-PFCRM accommodates not only the nominal input–output characteristics but also the noise statistics of the given data.

3.1.3. FM-PFCRM algorithm

Assume that a set of training data $\{\mathbf{x}_i, y_i\}_{i=1}^N$ is obtained from experiments and that we build a fuzzy model with c fuzzy rules to represent the data set.

Step 1: Assume c hyperplanes as initial cluster representatives:

$$y^r(\mathbf{x}) = a_0^r + a_1^r x_1 + \cdots + a_d^r x_d = \mathbf{x}^T \mathbf{a}_0^r, \quad (r = 1, \ldots, c)$$

The subscript "0" in $\mathbf{a}_0^r$ and σ_0^r means that this is the initial (the iteration index $j = 0$) step. Based on our experience, the authors set initial membership value $h_0^r(\mathbf{x}_i)$ by applying FCM to only input space and determine $\mathbf{a}_0^r$ and σ_0^r by (20) and (18), respectively.

Step 2 (*F-step*): We increment the iteration index ($j = j + 1$). At the jth iteration, we assign fuzzily each sample pair $(\mathbf{x}_i, y_i)$ to all clusters and compute a $c \times N$ membership matrix

$$H_j = \begin{pmatrix} h_j^1(\mathbf{x}_1) & h_j^1(\mathbf{x}_2) & \cdots & h_j^1(\mathbf{x}_N) \\ h_j^2(\mathbf{x}_1) & h_j^2(\mathbf{x}_2) & \cdots & h_j^2(\mathbf{x}_N) \\ \vdots & \vdots & \vdots & \vdots \\ h_j^c(\mathbf{x}_1) & h_j^c(\mathbf{x}_2) & \cdots & h_j^c(\mathbf{x}_N) \end{pmatrix}$$

by,

$$h_j^r(\mathbf{x}_i) = \frac{p\left(y_i|\mathbf{x}_i, \mathbf{\Theta}_{j-1}^r\right)}{\sum_{q=1}^{c} \left\{ p\left(y_i|\mathbf{x}_i, \mathbf{\Theta}_{j-1}^q\right) \right\}} = \frac{\left(\frac{1}{\sqrt{2\pi\left(\sigma_{j-1}^r\right)^2}} \exp\left(-\frac{1}{2} \frac{\left(y_i - y_{j-1}^r(\mathbf{x}_i)\right)^2}{\left(\sigma_{j-1}^r\right)^2} \right) \right)}{\sum_{q=1}^{c} \left(\frac{1}{\sqrt{2\pi\left(\sigma_{j-1}^q\right)^2}} \exp\left(-\frac{1}{2} \frac{\left(y_i - y_{j-1}^q(\mathbf{x}_i)\right)^2}{\left(\sigma_{j-1}^q\right)^2} \right) \right)} \tag{21}$$

where $\mathbf{\Theta}_{j-1}^r$ and σ_{j-1}^r denotes the parameters of the fuzzy model at the previous ($j-1$th) iteration.

Step 3 (*M-step*): Compute the new cluster representatives $\mathbf{a}^r$ and noise parameters σ^r at the jth iteration by

$$\mathbf{a}_j^r = \left(\sum_{i=1}^{N} \left(h_j^r(\mathbf{x}_i)\mathbf{x}_i\mathbf{x}_i^T \right) \right)^{-1} \left(\sum_{i=1}^{N} \left(h_j^r(\mathbf{x}_i) y_i \mathbf{x}_i \right) \right) \tag{22}$$

and

$$\sigma_j^r = \sqrt{\frac{\sum_{i=1}^{N} \left\{ h_j^r(\mathbf{x}_i) \left(y_i - y_j^r(\mathbf{x}_i) \right)^2 \right\}}{\sum_{i=1}^{N} h_j^r(\mathbf{x}_i)}}. \tag{23}$$

The cluster centers can be computed recursively by weighted recursive least square (WRLS), which are explained in detail in Appendix. This recursive implementation is useful especially when the number of data is small and the inverse matrix of (22) is almost singular.

Step 4: If $\|H_j - H_{j-1}\| \leqslant \varepsilon$, then stop and set $\mathbf{a}^r = \mathbf{a}_j^r$ and $\sigma^r = \sigma_j^r$; otherwise go to Step 2.

The proposed FM-PFCRM looks similar to the FCRM used in the previous paper [16,17]. It is interesting to see that two algorithms which started from completely different ideas lead to the same form. But the two algorithms differ in the way

each data point is assigned to a fuzzy rule in (15). In the FCRM, the inverse of the modeling error is used to assign the data points and the distribution of each fuzzy rule is not considered. In the FM-PFCRM, however, not only the modeling error but also the variance of each fuzzy rule is utilized to assign the data points according to the distribution of each fuzzy rule. After the consequent parameters $\mathbf{a}^r$ and the noise parameters σ^r are determined by FM-PFCRM, we compute the premise parameters of the membership functions $A_k^r(x) = \exp\left\{-\left(\frac{x-p_{k1}^r}{p_{k2}^r}\right)^2\right\}$where

$$p_{k1}^r = \frac{\sum_{i=1}^N h^r(\mathbf{x}_i) x_{i,k}}{\sum_{i=1}^N h^r(\mathbf{x}_i)} \tag{24}$$

and

$$p_{k2}^r = \sqrt{2}\sqrt{\frac{\sum_{i=1}^N h^r(\mathbf{x}_i)\left(x_{i,k} - p_{k1}^r\right)^2}{\sum_{i=1}^N h^r(\mathbf{x}_i)}}. \tag{25}$$

3.2. Fine tuning

In the previous coarse tuning, we partitioned the input space into several clusters and built the rough approximation to the unknown system with noise. In this subsection, we refine the parameters of the fuzzy model by finely adjusting the parameters. We call this step as the fine tuning. In this paper, a new fine tuning algorithm is proposed and it is completely different from those of the previous methods [16,17,26]. Since the previous fuzzy models were deterministic, the parameters were fine tuned by minimizing the approximation error. In the proposed method, however, the fuzzy model is probabilistic and the parameters are fine tuned by maximizing the likelihood of the given data points. More specifically, we consider the full likelihood function

$$L(\mathbf{\Theta}) = \log p(\mathbf{Y}|\mathbf{X},\mathbf{\Theta}) = \sum_{i=1}^N \left\{ \log\left(\frac{1}{\sqrt{2\pi\sigma_{fuz}^2(\mathbf{x}_i)}} \exp\left(-\frac{1}{2} \frac{\left(y_i - y_{fuzmean}(\mathbf{x}_i)\right)^2}{\sigma_{fuz}^2(\mathbf{x}_i)} \right) \right) \right\} \tag{26}$$

and maximize it by adjusting the parameters by the gradient ascent

$$\Delta\mathbf{\Theta} = \eta_\Theta \frac{\partial L(\mathbf{\Theta})}{\partial \mathbf{\Theta}}. \tag{27}$$

Using (8) and (9), the full log likelihood function (26) can be rewritten into

$$\begin{aligned}
L(\mathbf{\Theta}) = \log p(\mathbf{Y}|\mathbf{X},\mathbf{\Theta}) &= \sum_{i=1}^N \left\{ \log\left(\frac{1}{\sqrt{2\pi\left(\sum_{r=1}^C h^{r2}(\mathbf{x}_i)\sigma^{r2}\right)}} \exp\left(-\frac{1}{2} \frac{\left(y_i - \sum_{r=1}^C h^r(\mathbf{x}_i)\left(\mathbf{a}^{rT}\mathbf{x}_i\right)\right)^2}{\sum_{r=1}^C h^{r2}(\mathbf{x}_i)\sigma^{r2}} \right) \right) \right\} \\
&= \sum_{i=1}^N \left\{ -\frac{1}{2}\log\left(2\pi\sum_{r=1}^C h^{r2}(\mathbf{x}_i)\sigma^{r2}\right) - \frac{1}{2}\frac{\left(y_i - \sum_{r=1}^C h^r(\mathbf{x}_i)\left(\mathbf{a}^{rT}\mathbf{x}_i\right)\right)^2}{\sum_{r=1}^C h^{r2}(\mathbf{x}_i)\sigma^{r2}} \right\} \\
&= \sum_{i=1}^N \left\{ -\frac{1}{2}\log 2\pi - \frac{1}{2}\log\left(\frac{\sum_{r=1}^C \varpi^{r2}(\mathbf{x}_i)\sigma^{r2}}{\left(\sum_{q=1}^C \varpi^q(\mathbf{x}_i)\right)^2} \right) - \frac{1}{2}\frac{\left(\sum_{r=1}^C h^r(\mathbf{x}_i)\left(y_i - \mathbf{a}^{rT}\mathbf{x}_i\right)\right)^2}{\frac{\sum_{r=1}^C \varpi^{r2}(\mathbf{x}_i)\sigma^{r2}}{\left(\sum_{q=1}^C \varpi^q(\mathbf{x}_i)\right)^2}} \right\} \\
&= const + \sum_{i=1}^N \left\{ -\frac{1}{2}\log\left(\sum_{r=1}^C \varpi^{r2}(\mathbf{x}_i)\sigma^{r2}\right) + \log\left(\sum_{q=1}^C \varpi^q(\mathbf{x}_i)\right) - \frac{1}{2}\frac{\left(\frac{\sum_{r=1}^C \varpi^r(\mathbf{x}_i)\left(y_i - \mathbf{a}^{rT}\mathbf{x}_i\right)}{\sum_{q=1}^C \varpi^q(\mathbf{x}_i)}\right)^2}{\frac{\sum_{r=1}^C \varpi^{r2}(\mathbf{x}_i)\sigma^{r2}}{\left(\sum_{q=1}^C \varpi^q(\mathbf{x}_i)\right)^2}} \right\} \\
&= const + \sum_{i=1}^N \left\{ -\frac{1}{2}\log\left(\sum_{q=1}^C \varpi^{q2}(\mathbf{x}_i)\sigma^{q2}\right) + \log\left(\sum_{q=1}^C \varpi^q(\mathbf{x}_i)\right) - \frac{1}{2}\frac{\left(\sum_{q=1}^C \varpi^q(\mathbf{x}_i)\left(y_i - \mathbf{a}^{qT}\mathbf{x}_i\right)\right)^2}{\sum_{q=1}^C \varpi^{q2}(\mathbf{x}_i)\sigma^{q2}} \right\}
\end{aligned} \tag{28}$$

From (27), the premise parameters of the fuzzy model can be updated in the fine tuning by

$$\begin{aligned}\Delta p_{kl}^r &= \eta_p \frac{\partial L}{\partial p_{kl}^r} = \eta_p \sum_{i=1}^{N} \left\{ -\frac{\varpi^r(\mathbf{x}_i)\sigma^{r2}}{\sum_{q=1}^{C}\varpi^{q2}(\mathbf{x}_i)\sigma^{q2}} \frac{\partial \varpi^r}{\partial p_{kl}^r} + \frac{1}{\sum_{q=1}^{C}\varpi^{q}(\mathbf{x}_i)} \frac{\partial \varpi^r}{\partial p_{kl}^r} \right. \\ &\quad - \frac{1}{2} \frac{2\left(\sum_{q=1}^{C}\varpi^{q}(\mathbf{x}_i)\left(y_i - \mathbf{a}^{qT}\mathbf{x}_i\right)\right)\left(y_i - \mathbf{a}^{rT}\mathbf{x}_i\right)\frac{\partial \varpi^r}{\partial p_{kl}^r}\left(\sum_{q=1}^{C}\varpi^{q2}(\mathbf{x}_i)\sigma^{r2}\right)}{\left(\sum_{q=1}^{C}\varpi^{q2}(\mathbf{x}_i)\sigma^{q2}\right)^2} \\ &\quad \left. - \frac{1}{2} \frac{-2\left(\sum_{q=1}^{C}\varpi^{q}(\mathbf{x}_i)\left(y_i - \mathbf{a}^{qT}\mathbf{x}_i\right)\right)^2 \varpi^r(\mathbf{x}_i)\sigma^{r2}\frac{\partial \varpi^r}{\partial p_{kl}^r}}{\left(\sum_{q=1}^{C}\varpi^{q2}(\mathbf{x}_i)\sigma^{q2}\right)^2} \right\} \\ &= \eta_p \sum_{i=1}^{N} \left\{ \frac{1}{\sum_{q=1}^{C}\varpi^{q}(\mathbf{x}_i)} - \frac{\varpi^r(\mathbf{x}_i)\sigma^{r2} + \left(\sum_{q=1}^{C}\varpi^{q}(\mathbf{x}_i)\left(y_i - \mathbf{a}^{qT}\mathbf{x}_i\right)\right)\left(y_i - \mathbf{a}^{rT}\mathbf{x}_i\right)}{\sum_{q=1}^{C}\varpi^{q2}(\mathbf{x}_i)\sigma^{q2}} \right. \\ &\quad \left. + \frac{\left(\sum_{q=1}^{C}\varpi^{q}(\mathbf{x}_i)\left(y_i - \mathbf{a}^{qT}\mathbf{x}_i\right)\right)^2 \varpi^r(\mathbf{x}_i)\sigma^{r2}}{\left(\sum_{q=1}^{C}\varpi^{q2}(\mathbf{x}_i)\sigma^{q2}\right)^2} \right\} \frac{\partial \varpi^r}{\partial p_{kl}^r} \end{aligned} \tag{29}$$

where η_p is the learning rate for the premise parameters. If we use the minimum operator as t-norm

$$\varpi^r = t_{k=1}^{d}\left\{A_k^r(x_k)\right\} = \min_{k=1,2,\ldots,d} A_k^r(x_k) \tag{30}$$

then

(a) for $k = k^* \triangleq \arg_k \min_{k=1,2,\ldots,d} A_k^r(x_{ik})$

$$\frac{\partial \varpi^r}{\partial p_{k1}^r} = \frac{2}{p_{k2}^r} \frac{x_{ik} - p_{k1}^r}{p_{k2}^r} \exp\left\{ -\left(\frac{x_{ik} - p_{k1}^r}{p_{k2}^r}\right)^2 \right\} \tag{31}$$

$$\frac{\partial \varpi^r}{\partial p_{k2}^r} = \frac{x_{ik} - p_{k1}^r}{p_{k2}^r} \frac{\partial \varpi^r}{\partial p_{k1}^r}, \tag{32}$$

(b) for $k \neq k^*$

$$\frac{\partial \varpi^i}{\partial p_{k1}^r} = \frac{\partial \varpi^i}{\partial p_{k2}^r} = 0. \tag{33}$$

In the similar way, the consequent parameters can be updated by

$$\Delta a_k^r = \zeta \frac{\partial L}{\partial a_k^r} = \eta_a \sum_{i=1}^{N} \left\{ \frac{\left(\sum_{q=1}^{C}\varpi^{q}(\mathbf{x}_i)\left(y_i - \mathbf{a}^{qT}\mathbf{x}_i\right)\right)\varpi^r(\mathbf{x}_i)x_{ik}}{\sum_{q=1}^{C}\varpi^{q2}(\mathbf{x}_i)\sigma^{q2}} \right\}. \tag{34}$$

Then, the probabilistic parameters can be evaluated by

$$\sigma^r = \sqrt{\frac{\sum_{i=1}^{N}\left\{h^r(\mathbf{x}_i)\left(y_i - y_{fuzmean}(\mathbf{x}_i)\right)^2\right\}}{\sum_{i=1}^{N} h^r(\mathbf{x}_i)}} \tag{35}$$

where η_a is the learning rate for the consequent parameters and $h^r = \frac{\varpi^r}{\sum_{r=1}^{c}\varpi^r}$. The fine tuning algorithm is summarized as follows:

3.2.1. Fine tuning

Step 1: Initialize the fuzzy model parameter $\boldsymbol{\Theta}$

$$\boldsymbol{\Theta} = \begin{pmatrix} \boldsymbol{\Theta}^1 & \boldsymbol{\Theta}^2 & \cdots & \boldsymbol{\Theta}^c \end{pmatrix}$$

$$\boldsymbol{\Theta}^r = \left(\underbrace{p_{11}^r \quad p_{12}^r \quad p_{21}^r \quad p_{22}^r \quad \cdots \quad p_{d1}^r \quad p_{d2}^r}_{\text{Premise variables}} \underbrace{a_0^r \quad a_1^r \quad \cdots \quad a_d^r}_{\text{Consequent variables } \mathbf{a}^r} \underbrace{\sigma^r}_{\text{probabilistvariable}} \right)$$

as the one obtained in the coarse tuning.

Step 2: For a sample pair $(\mathbf{x}_i, y_i)$ with nonzero $\sum_{r=1}^{c} h^r(\mathbf{x}_i)$, we compute the fuzzy model $y_{fuzmean}(\mathbf{x}_i)$ and update the parameter $\boldsymbol{\Theta}$ by

$$\Delta p_{kl}^r = \eta_p \sum_{i=1}^{N} \left\{ \frac{1}{\sum_{q=1}^{C} \varpi^q(\mathbf{x}_i)} - \frac{\varpi^r(\mathbf{x}_i)\sigma^{r2} + \left(\sum_{q=1}^{C} \varpi^q(\mathbf{x}_i)\left(y_i - \mathbf{a}^{qT}\mathbf{x}_i\right)\right)\left(y_i - \mathbf{a}^{rT}\mathbf{x}_i\right)}{\sum_{q=1}^{C} \varpi^{q2}(\mathbf{x}_i)\sigma^{q2}} + \frac{\left(\sum_{q=1}^{C} \varpi^q(\mathbf{x}_i)\left(y_i - \mathbf{a}^{qT}\mathbf{x}_i\right)\right)^2 \varpi^r(\mathbf{x}_i)\sigma^{r2}}{\left(\sum_{q=1}^{C} \varpi^{q2}(\mathbf{x}_i)\sigma^{q2}\right)^2} \right\} \frac{\partial \varpi^r}{\partial p_{kl}^r} \tag{36}$$

$$\Delta a_k^r = \zeta \frac{\partial L}{\partial a_k^r} = \eta_a \sum_{i=1}^{N} \left\{ \frac{\left(\sum_{q=1}^{C} \varpi^q(\mathbf{x}_i)\left(y_i - \mathbf{a}^{qT}\mathbf{x}_i\right)\right)\varpi^r(\mathbf{x}_i)x_{ik}}{\sum_{q=1}^{C} \varpi^{q2}(\mathbf{x}_i)\sigma^{q2}} \right\}. \tag{37}$$

Step 3: Update the probabilistic parameter

$$\sigma^r = \sqrt{\frac{\sum_{i=1}^{N}\left\{h^r(\mathbf{x}_i)\left(y_i - y_{fuzmean}(\mathbf{x}_i)\right)^2\right\}}{\sum_{i=1}^{N} h^r(\mathbf{x}_i)}}. \tag{38}$$

Step 4: Compute the full log likelihood

$$L(\Theta) = \sum_{i=1}^{N} \left\{ \log\left(\frac{1}{\sqrt{2\pi\sigma_{fuz}^2(\mathbf{x}_i)}} \exp\left(-\frac{1}{2} \frac{\left(y_i - y_{fuzmean}(\mathbf{x}_i)\right)^2}{\sigma_{fuz}^2(\mathbf{x}_i)} \right) \right) \right\}. \tag{39}$$

Step 5: Check for convergence of the likelihood or the parameters. If the convergence is not reached, go to Step 2; otherwise terminate the fine tuning.

4. Simulation results

In this section, we apply the proposed probabilistic fuzzy model to numerical examples and compare them with the previous methods to demonstrate the validity of the proposed method. In all the simulations, we use 5×2 cv Dietterich's test [28,32] to show the statistical validity of the algorithms. More specifically, when a data set is given, the data points are randomly permuted, the first half of the data points is used as a training set and the second half is used as a test set to assess the modeling methods. This is repeated for five different random permutations.

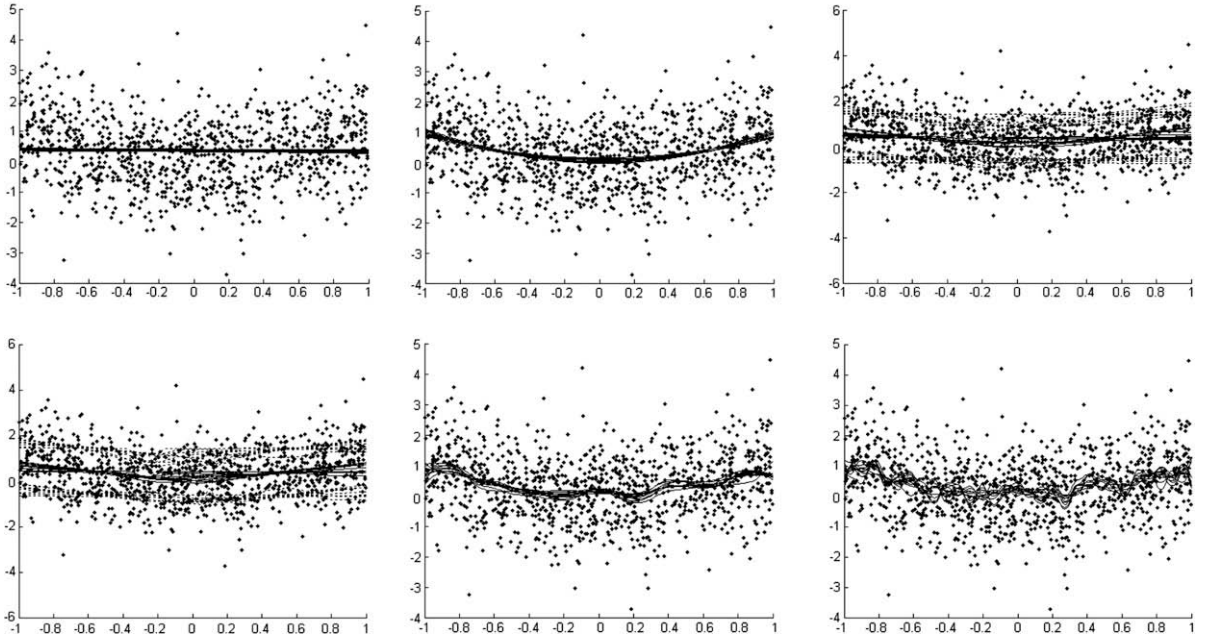


Fig. 3. 1D quadratic function with uniform noise. From left to right, first row: linear, quadratic (optimal), fuzzy model with three rules (FM3), FM5, second row: MLP with 10 hidden nodes (MLP10), MLP30, third row: MLP 50, RSB with five memberships for input and five memberships for output (RSB5×5), RSB10×10, fourth row: WM3×3, WM5×5, WM10×10.

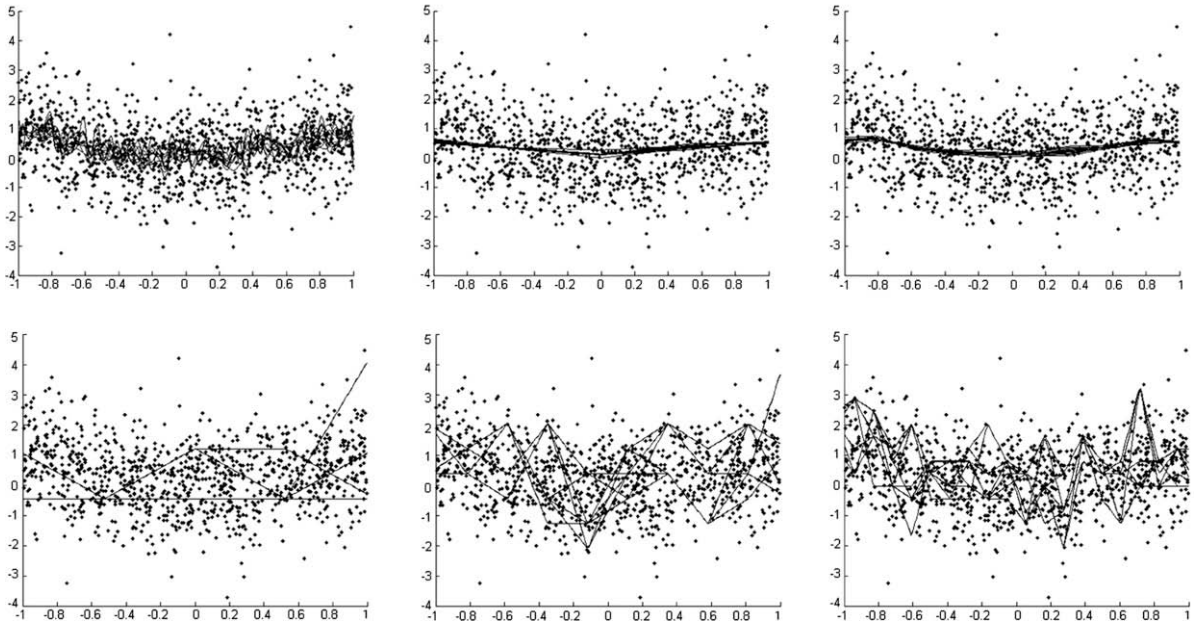


Fig. 3 (*continued*)

Table 1
Test error for 1D quadratic function with uniform noise.

	Mean	Median		Mean	Median
FM3	1.057	1.0502	QUAD	1.0363	1.0354
FM5	1.0709	1.0763	RSB5	1.0581	1.0591
LIN	1.1058	1.1069	RSB10	1.0466	1.0497
MLP10	1.0414	1.038	WM5	1.6363	1.6123
MLP30	1.0569	1.0579	WM10	1.8425	1.912
MLP50	1.0972	1.0951	WM20	1.6929	1.5969

4.1. 1D quadratic function with uniform noise

1-Dimensional quadratic function

$$y = x^2 + \varepsilon \tag{40}$$

is considered, where $\varepsilon \sim N(\cdot|0, 1^2)$. This example is taken from [28]. 2000 data points are sampled from (40) and 5×2 cv test is conducted. For the sake of comparison, a linear model, a quadratic model, a multilayer perceptron (MLP) neural network [33], Wang and Mendel's (WM) fuzzy model [3] are applied as deterministic models and a random set based (RSB) fuzzy

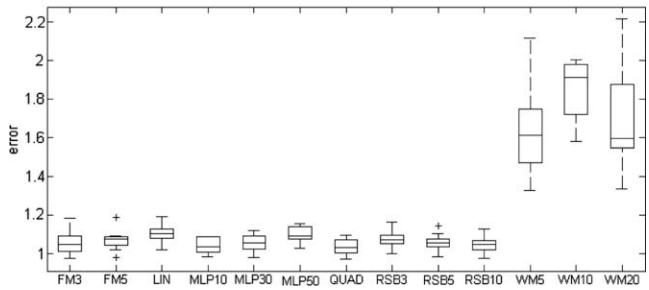


Fig. 4. Boxplot for 1D quadratic function with uniform noise. It denotes the minimum and maximum modeling error, the median, 25% and 75% percentiles of 5×2 cv.

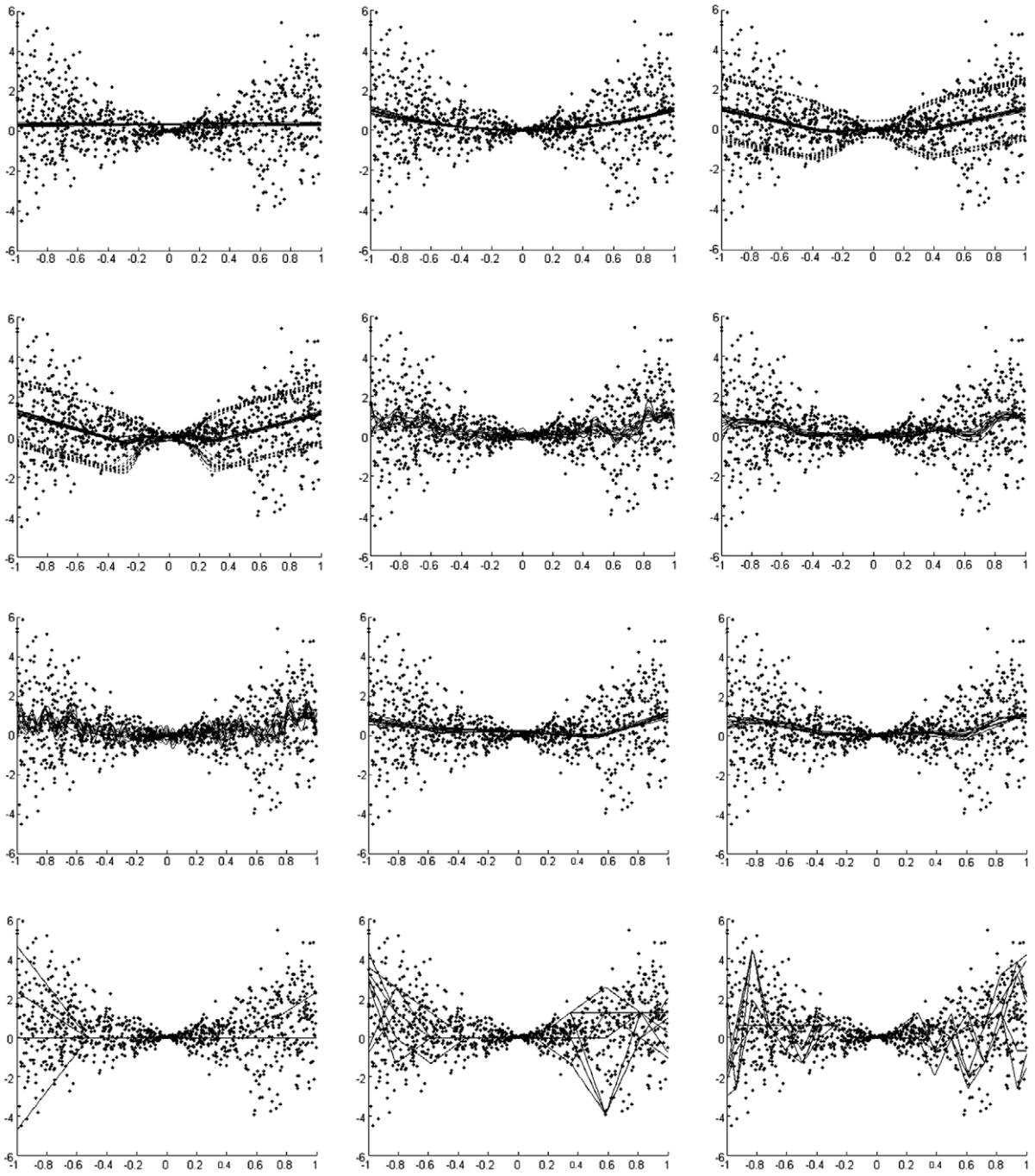


Fig. 5. 1D quadratic function with state-dependent noise. From left to right, first row: linear, quadratic (optimal), fuzzy model with three rules (FM3), FM5, second row: MLP with 10 hidden nodes (MLP10), MLP30, third row: MLP 50, RSB with five memberships for input and five memberships for output (RSB5×5), RSB10×10, fourth row: WM3×3, WM5×5, WM10×10.

model [28] is applied as a probabilistic model. Fig. 3 shows the modeling results of the proposed and the previous methods for the 5×2 cv test. In each figure, ten modeling results obtained from 5×2 cv are depicted in solid lines and training data points are denoted as dots.

Unlike the previous methods, the proposed probabilistic method also models the statistical variance of the data and it is depicted in a dotted line. It can be seen that the probabilistic characteristic is well approximated by the proposed fuzzy model. Simulation results are summarized in Table 1 and Fig. 4.

Table 2
Test error for 1D quadratic function with state-dependent noise.

	Mean	Median		Mean	Median
FM3	1.8158	1.8182	QUAD	1.8123	1.8122
FM5	1.8337	1.8348	RSB5	1.8279	1.83
LIN	1.9011	1.8995	RSB10	1.8169	1.8244
MLP10	1.8186	1.8261	WM5	2.8715	3.0407
MLP30	1.848	1.8625	WM10	3.0204	3.0535
MLP50	1.8809	1.8719	WM20	2.8134	2.8271

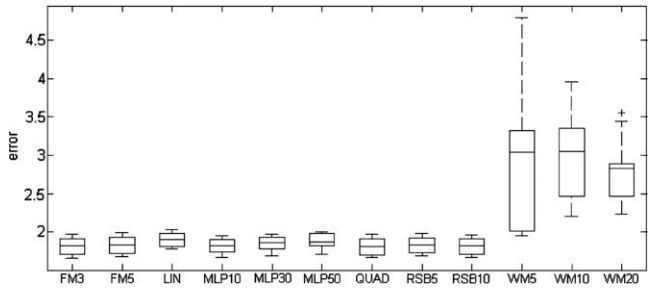


Fig. 6. Boxplot for 1D quadratic function with state-dependent noise.

As expected, quadratic model shows the best performance in this case. FM, RSB, and MLP are good enough. As the complexity of the model increases (the number of membership functions or the number of rules), the FM and the MLP exhibit the slight over fitting but the RSB shows the consistent results.

4.2. 1D quadratic function with input-dependent noise

This example is very similar to the previous quadratic model but the statistics of the added noise depends on the input variable:

$$y = x^2 + \varepsilon \tag{41}$$

where $\varepsilon \sim N(\cdot\,|0, 5x^2)$. As in the previous example, 2000 data points are sampled from (41) and 5×2 cv test is conducted. In Fig. 5, the proposed method is compared with the previous methods.

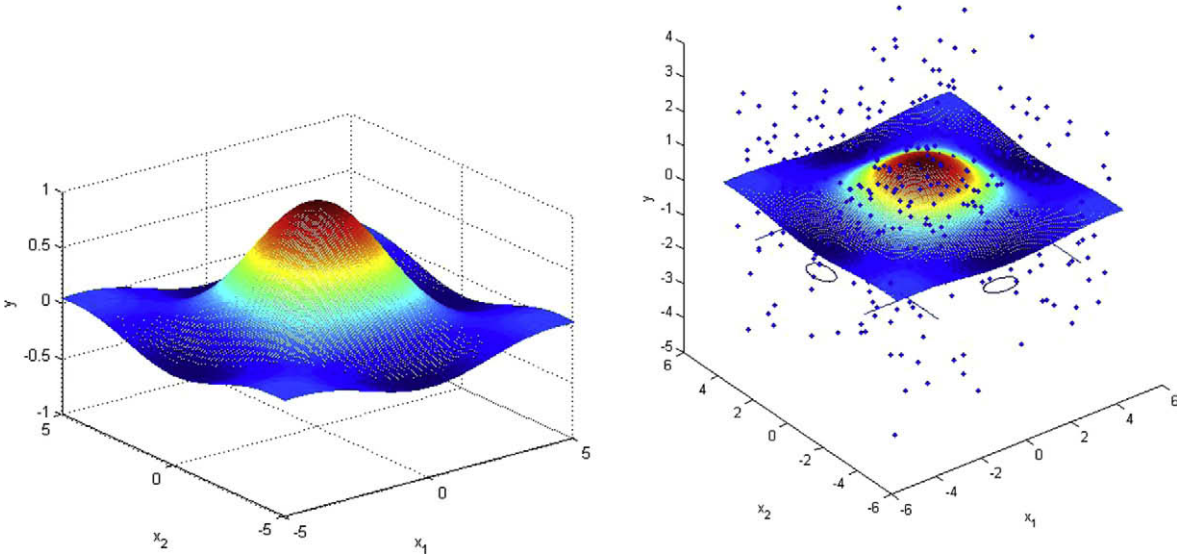


Fig. 7. 2D sinc function (left) and a training set (right).

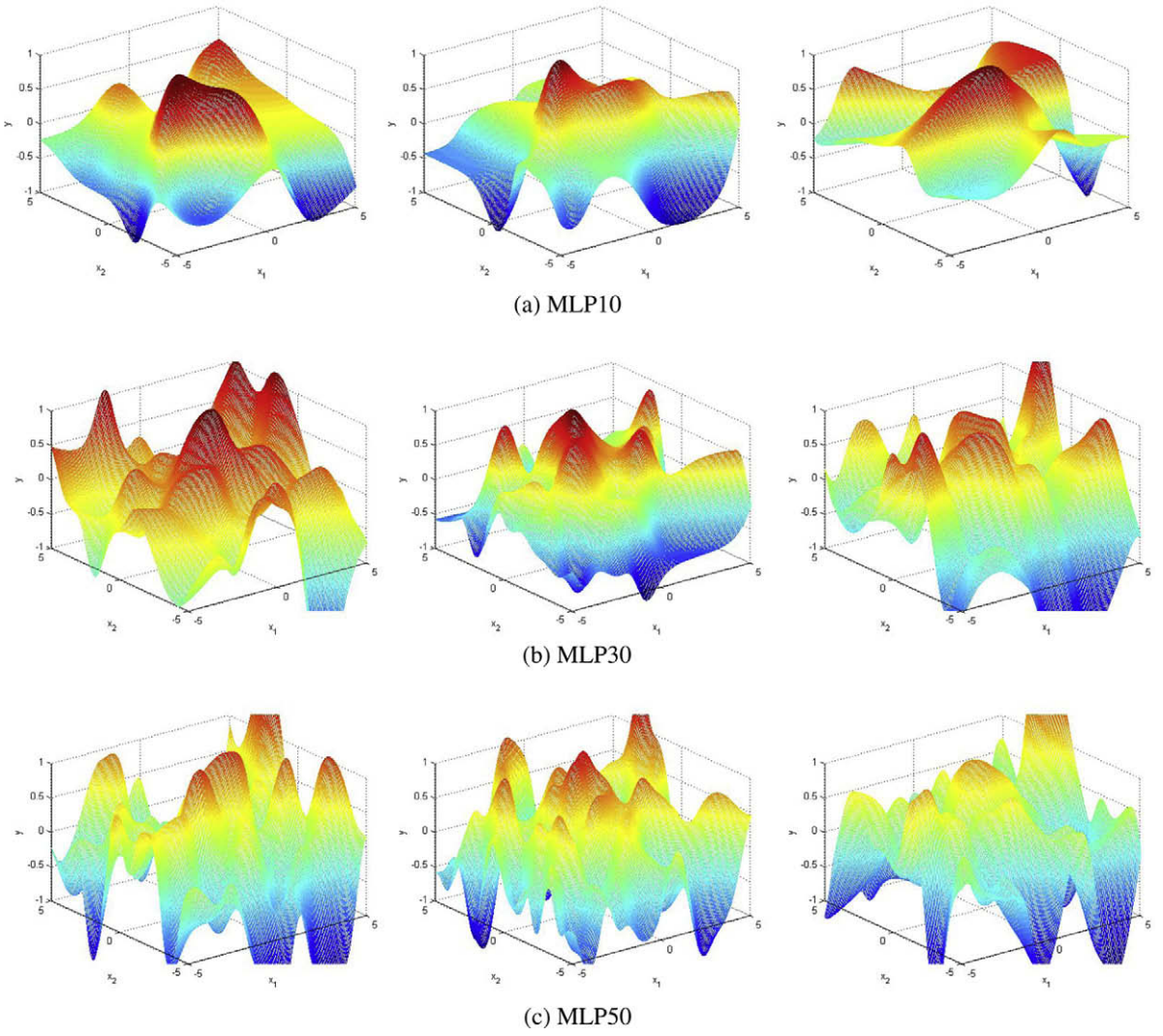


Fig. 8. From left to right columns: the median, best, worst. From the top to bottom rows: MLP10, MLP30, MLP50, WM5×5, WM7×7, WM10×10, RSB5×5, RSB7×7, FM.

Overall, the similar results are obtained as in the previous example. The statistical variance of the data is depicted in a dotted line in the proposed method (FM3 and FM5). It can be observed that the noise model is also well approximated by the proposed fuzzy model. Simulation results are summarized in Table 2 and Fig. 6.

In this example, the differences between the best and worst performances in all the methods are larger than those of the previous example with uniform noise. As in the previous example, the quadratic model shows the best performance and FM, RSB, and MLP are also good enough.

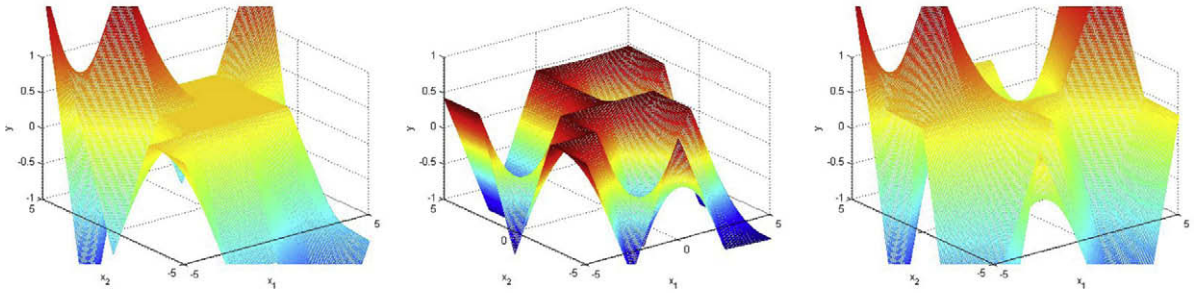
4.3. 2D sinc function with input-dependent noise

The third example is a two-variable sinc function

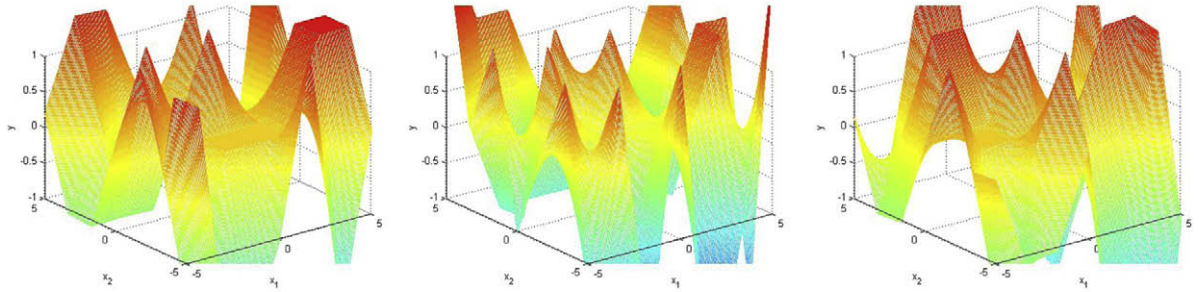
$$y = \frac{\sin(x_1)\sin(x_2)}{x_1 x_2} + \varepsilon, \quad -5 \leqslant x_1 \leqslant 5, \quad -5 \leqslant x_2 \leqslant 5\,. \tag{42}$$

where $\varepsilon \sim N(\cdot|0, \frac{1}{10}(x_1^2 + x_2^2))$. This is similar to the example in [26] but the input-dependent noise ε. 1000 data points are sampled from (42). 500 points are used for the training and the remaining 500 points are used for the test. Fig. 7 shows the noise-free sinc function and the data points superimposed on it.

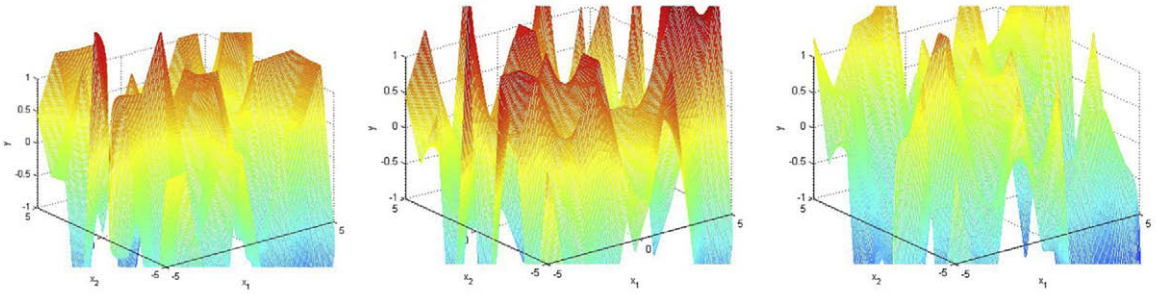
Since the noise is much larger than the nominal function, the two subfigures in Fig. 7 use the different intervals for y axis. 5×2 cv test is conducted. Fig. 8 compares the modeling result of the proposed and the previous methods for the 2D sinc



(d) WM5x5



(e) WM7x7



(f) WM10x10

Fig. 8 (*continued*)

function. In the figure, left, middle and right columns show the modeling result of the median, best and worst cases of each method, respectively.

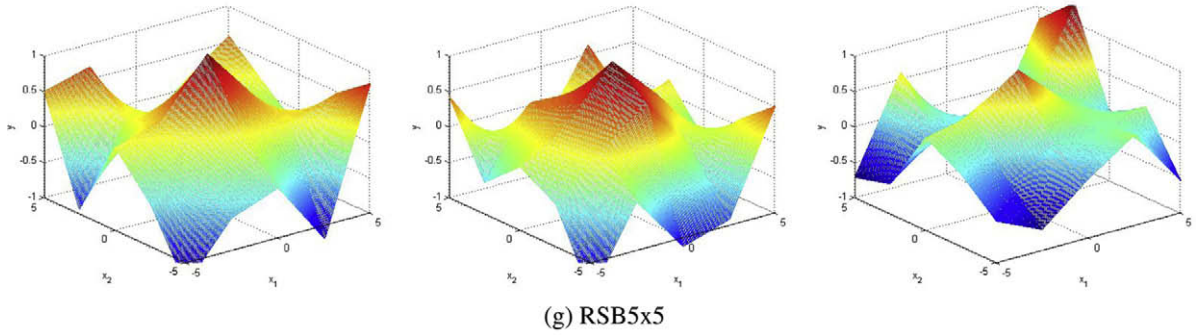
As can be seen from the figure, the approximation models of the previous algorithms are seriously distorted from the given sinc function but the proposed method exhibits the excellent modeling performance in this example. Fig. 9 shows the statistical variance approximated by the proposed fuzzy model.

From the figure, it can be seen that, although distorted from the ground truth $\varepsilon \sim N(\cdot|0, \frac{1}{10}(x_1^2 + x_2^2))$, the approximated variance has the shape of the quadratic function. Simulation results for 2D sinc function are summarized in Table 3 and Fig. 10.

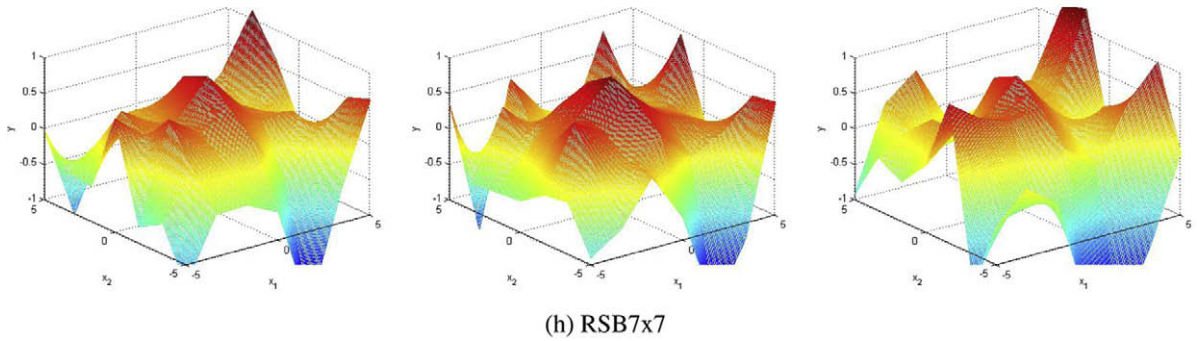
In this example, the probabilistic methods such as FM and RSB outperform the deterministic methods such as MLP or WM. Especially, the proposed fuzzy model with six rules exhibit the best and the most consistent approximation performance in the presence of the input-dependent error.

4.4. Concrete compressive strength data set

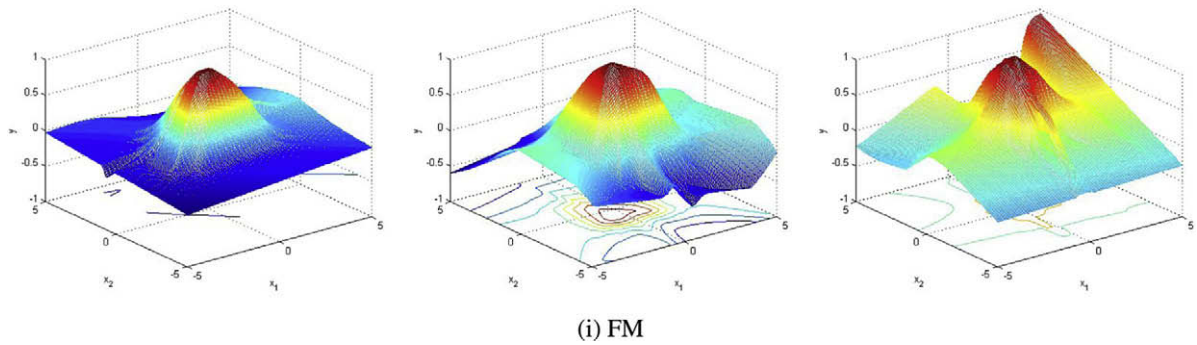
This dataset is taken from UCI machine learning repository [34,35]. It is a real world problem about the concrete compressive strength and it is important in civil engineering. This dataset has eight input variables and 1 output variable summarized as follows: Input: (1) Cement, (2) Blast Furnace Slag, (3) Fly Ash, (4) Water, (5) Superplasticizer, (6) Coarse Aggregate, (7) Fine Aggregate, (8) Age – Day (1–365); Output: Concrete compressive strength. 5×2 cv test is conducted and simulation results are summarized in Table 4 and Fig. 11.



(g) RSB5x5



(h) RSB7x7



(i) FM

Fig. 8 (*continued*)

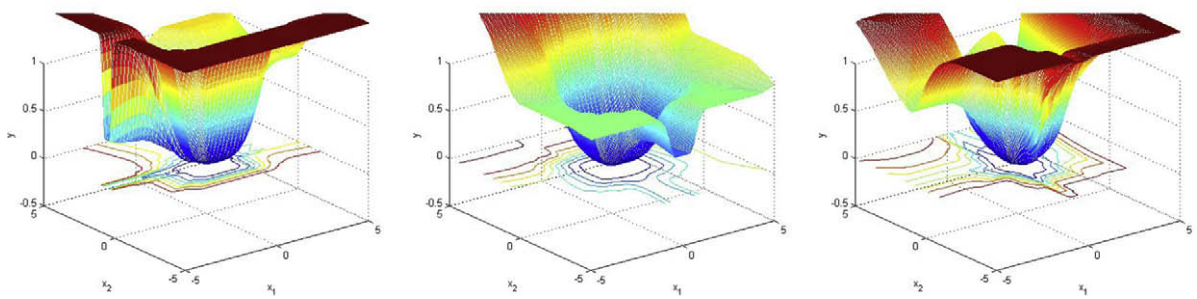


Fig. 9. The variance of the data approximated by the fuzzy model: from left to right, median, best, worst.

In this case, the MLP is trained not by the gradient descent back propagation with adaptive learning rate but by the Levenberg–Marquardt since the training by the gradient descent exhibits poor performance. The MLP trained by the Levenberg–Marquardt shows the best performance among all the algorithms but it is easily overfitted and its performance

Table 3
Test error for 2D sinc function.

	Mean	Median		Mean	Median
FM6	1.5553	1.556	RSB5	1.5971	1.5883
FM9	1.5683	1.5723	RSB7	1.6285	1.6074
MLP10	1.6103	1.606	RSB10	1.6829	1.7151
MLP30	1.672	1.6741	WM5	2.1908	2.1927
MLP50	1.7338	1.7233	WM7	2.201	2.1822
MLP100	1.8452	1.8329	WM10	2.2361	2.2107

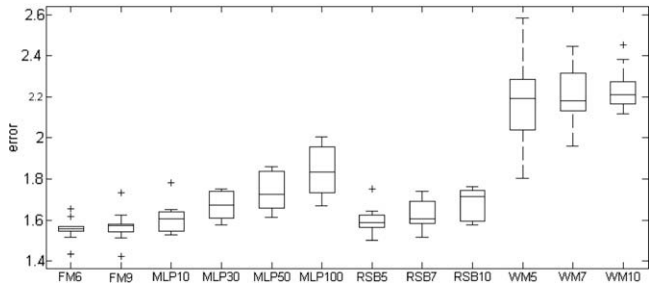


Fig. 10. Boxplot for 2D sinc function.

Table 4
Test error for original concrete compressive strength dataset.

	Mean	Median		Mean	Median
FM3	73.9065	73.1798	MLP50	94.4324	104.9658
FM5	75.5618	77.0354	RSB2×5	177.8328	177.627
FM10	70.0944	71.7341	WM2×5	203.512	204.1995
MLP5	87.8377	63.3774	WM3×5	179.6136	181.5768
MLP30	87.1124	72.8074	WM3×10	180.4718	183.753

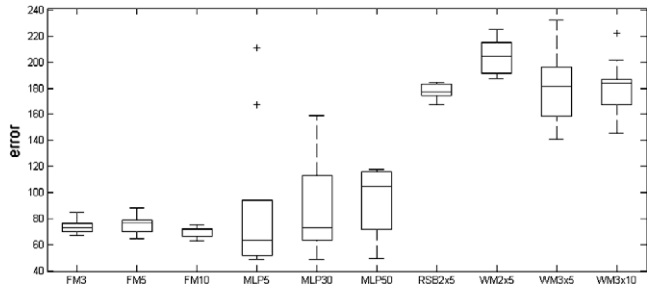


Fig. 11. Boxplot for original concrete compressive strength dataset.

is less consistent than others. The proposed FM exhibits the second best and the most consistent performance among all the algorithms. Further, to see the impact of the noise on the modeling problem, some noises represented by

$$\varepsilon \sim N\left(\cdot|0, 0.005 \times (\text{Blast Furnace Slag} - 150)^2 + 0.003 \times (\text{fly ash} - 150)^2\right) \tag{43}$$

are added. 5×2 cv test is conducted and simulation results are summarized in Table 5 and Fig. 12.

When the noise represented by (43) is added, the proposed fuzzy model outperforms the MLP and exhibits not only the most consistent but also the best performance among all the algorithms.

4.5. Building 2

This dataset is taken from PROBEN 1 [36]. It is also a real world problem about the prediction of energy consumption in a building. The hourly consumption of electrical energy should be predicted based on the date, the time of day, outside

Table 5
Test error for original concrete compressive strength dataset contaminated with noise.

	Mean	Median		Mean	Median
FM3	170.3792	171.778	MLP50	189.2188	195.9908
FM5	165.3242	165.3513	RSB2×5	266.0105	263.6762
FM10	162.8778	162.713	WM2×5	302.3741	306.362
MLP5	195.0595	188.0784	WM3×5	331.4997	333.0657
MLP30	195.3100	199.8669	WM3×10	330.1658	338.5157

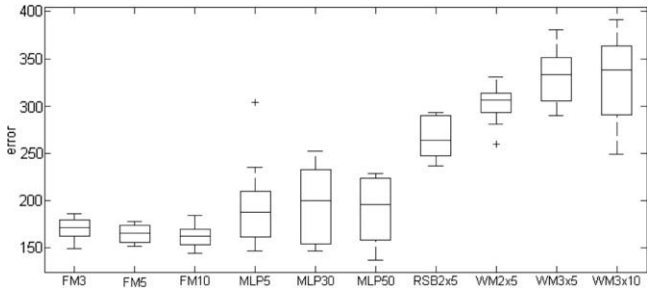


Fig. 12. Boxplot for concrete compressive strength dataset contaminated with noise.

temperature, outside humidity, solar radiations, etc. This dataset has 14 inputs, 3 outputs and 4208 examples. Only the first output is considered in this simulation. The amount of noise is very small. 5×2 cv test is conducted and simulation results are summarized in Table 6 and Fig. 13.

In this example, the MLP shows the best performance among the algorithms and the MLP5 exhibits very consistent result. This example shows that when the amount of noise is very small, the MLP can be the best choice for modeling. It is interesting to see that the WM also shows reasonably good performance in this example in contrast to [28] in which WM showed very poor performance. The reason is that more membership functions are used in this example than in [28]. From this example, it can be seen that when the amount of noise is small and linguistically understandable model is needed, WM could be a good choice. The proposed fuzzy model and the RSB exhibit the reasonably good and consistent performance.

Table 6
Test error for building 2.

	Mean	Median		Mean	Median
FM3	0.0031	0.0031	MLP50	0.0042	0.0042
FM5	0.0034	0.0035	RSB	0.0049	0.0049
LIN	0.0048	0.0048	WM2×5	0.0053	0.0053
MLP5	0.0029	0.0029	WM3×5	0.0035	0.0034
MLP10	0.0029	0.0029	WM3×10	0.0034	0.0034
MLP30	0.0029	0.0029			

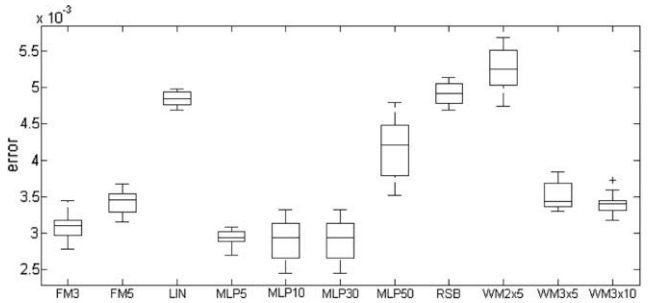


Fig. 13. Boxplot for building 2.

5. Conclusion

In this paper, a probabilistic fuzzy model has been proposed to approximate the sample data with noises and outliers. The new model consists of the nominal noise-free part and the probabilistic noise part. To identify the two parts simultaneously, the FM was proposed. The proposed model was applied to two examples and demonstrated its effectiveness by being compared with neural network.

Acknowledgements

This work was supported by Manpower Development Program for Energy & Resources of MKE with Yonsei Electric Power Research Center (YEPRC) at Yonsei University, Seoul, Korea. The corresponding author also appreciates Prof. Lotfi A. Zadeh for the facilities provided in BISC (Berkeley Initiative in Soft Computing) when the manuscript is prepared.

Appendix. Weighted recursive least square (WRLS) for fuzzy modeling

Suppose that a set of sample data $\{\mathbf{x}_i, y_i\}_{i=1}^N$ is given and the ith sample data belong to the rth cluster to the degree of $h_j^r(\mathbf{x}_i)$ where the subscript j denotes the number of iteration in FM-PFCRM. Then, the cluster representative $\mathbf{a}_j^r$ represented by

$$\mathbf{a}_j^r = \left(\sum_{i=1}^{N}\left(h_j^r(\mathbf{x}_i)\mathbf{x}_i\mathbf{x}_i^T\right)\right)^{-1}\left(\sum_{i=1}^{N}\left(h_j^r(\mathbf{x}_i)y_i\mathbf{x}_i\right)\right)$$

can be computed recursively over the given sample data ($i = 1,\ldots,N$) by

$$\mathbf{K}_{j,i}^r = \frac{\mathbf{S}_{j,i-1}^r\mathbf{x}_i^r}{\frac{1}{h_j^r(\mathbf{x}_i)} + \mathbf{x}_i^{rT}\mathbf{S}_{j,i-1}^r\mathbf{x}_i^r}$$

$$\mathbf{a}_{j,i}^r = \mathbf{a}_{j,i-1}^r + \mathbf{K}_{j,i}^r\left(y_i - \mathbf{x}_i^T\mathbf{a}_{j,i-1}^r\right)$$

$$\mathbf{S}_{j,i}^r = \left[\mathbf{I} - \mathbf{K}_{j,i}^r\mathbf{x}_i^T\right]\mathbf{S}_{j,i-1}^r$$

with the initial values $\mathbf{a}_{j,0}^r = \mathbf{0}$, and $\mathbf{S}_{j,0}^r = \alpha\mathbf{I}$ and final values $\mathbf{a}_j^r \triangleq \mathbf{a}_{j,N}^r$, where $\mathbf{I}$ is an identity matrix and α is a sufficiently large positive number.

References

[1] A.F. Gobi, W. Pedrycz, Fuzzy modelling through logic optimization, International Journal of Approximate Reasoning 45 (3) (2007) 488–510.
[2] L.-X. Wang, J.M. Mendel, Generating fuzzy rules by learning from examples, IEEE Transactions on Systems, Man and Cybernetics 22 (6) (1992) 1414–1427.
[3] L. delaOssa, J.A. Gámez, J.M. Puerta, Learning weighted linguistic fuzzy rules by using specifically-tailored hybrid estimation of distribution algorithms, International Journal of Approximate Reasoning 50 (3) (2009) 541–560.
[4] M. Sugeno, T. Yasukawa, A fuzzy-logic-based approach to qualitative modeling, IEEE Transactions on Fuzzy Systems 1 (1) (1993) 7–31.
[5] L.X. Wang, J.M. Mendel, Fuzzy basis functions, universal approximation, and orthogonal least squares learning, IEEE Transactions on Neural Networks 3 (5) (1992) 807–814.
[6] E. Kim, W. Pedrycz, Information granulation as a basis of fuzzy modeling, Journal of Intelligent & Fuzzy Systems 18 (2) (2007) 123–148.
[7] C.-T. Lin, Y.-C. Lu, A neural fuzzy system with linguistic teaching signals, IEEE Transactions on Fuzzy Systems 3 (2) (1995) 169–189.
[8] C.-T. Lin, W.-C. Cheng, S.-F. Liang, An on-line ICA-mixture-model-based self-constructing fuzzy neural network, IEEE Transactions on Circuits and Systems I: Fund. Theory and Applications 52 (1) (2005) 207–221.
[9] H. Ishibuchi, Y. Nojima, Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning, International Journal of Approximate Reasoning 44 (1) (2007) 4–31.
[10] J. González, I. Rojas, H. Pomares, L.J. Herrera, A. Guillén, J.M. Palomares, F. Rojas, Improving the accuracy while preserving the interpretability of fuzzy function approximators by means of multi-objective evolutionary algorithms, International Journal of Approximate Reasoning 44 (1) (2007) 32–44.
[11] R. Alcalá, J. Alcalá-Fdez, F. Herrera, J. Otero, Genetic learning of accurate and compact fuzzy rule based systems based on the 2-tuples linguistic representation, International Journal of Approximate Reasoning 44 (1) (2007) 45–64.
[12] E.V. Broekhoven, V. Adriaenssens, B.D. Baets, Interpretability-preserving genetic optimization of linguistic terms in fuzzy models for fuzzy ordered classification: An ecological case study, International Journal of Approximate Reasoning 44 (1) (2007) 65–90.
[13] T. Takagi, M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, IEEE Transactions on Systems Man and Cybernetics 15 (1) (1985) 116–132.
[14] M. Sugeno, G.T. Kang, Fuzzy modeling and control of multilayer incinerator, Fuzzy Sets and Systems 18 (1986) 329–346.
[15] M. Sugeno, G.T. Kang, Structure identification of a fuzzy model, Fuzzy Sets and Systems 28 (1988).
[16] E. Kim, M. Park, S. Ji, M. Park, A new approach to fuzzy modeling, IEEE Transactions on Fuzzy Systems 5 (3) (1997) 328–337.
[17] E. Kim, M. Park, S. Kim, M. Park, A transformed input-domain approach to fuzzy modeling, IEEE Transactions on Fuzzy Systems 6 (4) (1998) 596–604.
[18] J.-N. Choi, S.-K. Oh, W. Pedrycz, Structural and parametric design of fuzzy inference systems using hierarchical fair competition-based parallel genetic algorithms and information granulation, International Journal of Approximate Reasoning 49 (3) (2008) 631–648.
[19] L.F. Mendonça, S.M. Vieira, J.M.C. Sousa, Decision tree search methods in fuzzy modeling and classification, International Journal of Approximate Reasoning 44 (2) (2007) 106–123.
[20] S. Roh, W. Pedrycz, S.-K. Oh, Genetic Optimization of Fuzzy Polynomial Neural Networks, IEEE Transactions on Industrial Electronics 54 (4) (2007) 2219–2238.
[21] D. Kukolj, Design of adaptive Takagi–Sugeno–Kang fuzzy models, Applied Soft Computing 2 (2002) 89–103.
[22] D. Kukolj, E. Levi, Identification of complex systems based on neural and Takagi–Sugeno fuzzy model, IEEE Transactions on Systems, Man, and Cybernetics, Part B 34 (1) (2004) 272–282.

[23] S. Roh, W. Pedrycz, S.-K. Oh, Genetic optimization of fuzzy polynomial neural networks, IEEE Transactions on Industrial Electronics 54 (4) (2007) 2219–2238.
[24] L. Sanchez, Interval-valued GA-P algorithms, IEEE Transactions on Evolutionary Computation 4 (1) (2000) 64–72.
[25] C.-C. Chuang, S.-F. Su, C.-C. Hsiao, The annealing robust backpropagation (ARBP) learning algorithm, IEEE Transactions on Neural Networks 11 (5) (2000) 1067–1077.
[26] C.-C. Chuang, S.-F. Su, S.-S. Chen, Robust TSK fuzzy modeling for function approximation with outliers, IEEE Transactions on Fuzzy Systems 9 (6) (2001) 810–821.
[27] J. Kim, Y. Suga, S. Won, A new approach to fuzzy modeling of nonlinear dynamic systems with noise: relevance vector learning mechanism, IEEE Transactions on Fuzzy Systems 14 (2) (2006) 222–231.
[28] L. Sanchez, J. Casillas, O. Cordon, M.J. Del Jesus, Some relationships between fuzzy and random classifiers and models, International Journal on Approximate Reasoning 29 (2002) 175–213.
[29] L. Sanchez, A random sets-based method for identifying fuzzy models, Fuzzy Sets and Systems 98 (1998) 343–354.
[30] S. Thrun, W. Bugard, D. Fox, Probabilistic Robotics, MIT Press, Cambridge, MA, 2005.
[31] C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
[32] T.G. Dietterich, Approximate statistical tests for comparing supervised classification learning algorithms, Neural Computation 10 (7) (1998) 1895–1923.
[33] M.T. Hagan, H.B. Demuth, H. Beale, Neural Network Design, PWS Publishing Company, USA, 1995.
[34] I-C. Yeh, Modeling of strength of high performance concrete using artificial neural networks, Cement and Concrete Research 28 (12) (1998) 1797–1808.
[35] UC Irvine Machine Learning Repository, <http://archive.ics.uci.edu/ml/index.html>.
[36] L. Prechelt, PROBEN 1 – a set of benchmarks and benchmarking rules for neural network training algorithms, Technical report 21/94, Fakultät für Informatik, Universität Karlsruhe, 1994.